A HYBRID CHAOS FIREFLY ALGORITHM FOR THREE-DIMENSIONAL IRREGULAR PACKING PROBLEM

CHUANXIN ZHAO
School of Computer Sciences and information, Anhui Normal University
Anhui Provincial Key Laboratory of Network and Information Security
Wuhu, 241000, China

LIN JIANG* AND KOK LAY TEO
Department of Mathematics and Statistics, Curtin University
Perth, WA 6845, Australia

(Communicated by Chao Xu)

ABSTRACT. The packing problem study how to pack multiple objects without overlap. Various exact and approximate algorithms have been developed for two-dimensional regular and irregular packing as well as three-dimensional bin packing. However, few results are reported for three-dimensional irregular packing problems. This paper will develop a method for solving three-dimensional irregular packing problems. A three-grid approximation technique is first introduced to approximate irregular objects. Then, a hybrid heuristic method is developed to place and compact each individual objects where chaos search is embedded into firefly algorithm in order to enhance the algorithm’s diversity for optimizing packing sequence and orientations. Results from several computational experiments demonstrate the effectiveness of the hybrid algorithm.

1. Introduction. The cutting problem and packing problem have rich applications in the real world, especially in industrial engineering and logistics, such as sheet metal cutting, glass industry, and container loading. The purpose of this kind of problem is to minimize waste in sheet cutting or to minimize space usage in cargo loading. The two type problems are equivalent in nature, and are regularly considered together.

Due to its wide range of applications, the cutting and packing problem has been considered widely in the literature [6]. In general, packing problems can be classified into two-dimensional packing and three-dimensional packing. Most of the existing results focus on two-dimensional packing. For two-dimensional regular packing, details can be found in the survey paper [23]. In [26], an exact approach and a randomized metaheuristic are designed to pack smallest square into a set of rectangular items. Compared with regular packing, the irregular packing problem is much more

2010 Mathematics Subject Classification. Primary: 80M50; Secondary: 90C27.

Key words and phrases. Irregular packing, raster approximation, firefly algorithm, chaos search.

The first author is supported by NSFC grant (61871412,61772034,61572036,61672039,61473326), Anhui Provincial Natural Science Foundation(1708085MF156,1808085MF172), Australian Research Council Linkage Program LP140100873.

* Corresponding author: Lin Jiang.
complicated and is, in fact, equivalent to an NP-hard combinatorial optimization problem [6]. In [2], the two-dimensional irregular packing problem is formulated as a mixed integer optimization problem without overlapping constraints. A branch and bound algorithm is proposed to solve this mixed integer optimization problem. However, exact methods like branch and bound methods can only be applied to solve small scale combinatorial problems [1]. If suboptimal solutions are sufficient, the meta-heuristics and approximate approaches can be applied. Heuristic methods do not require the derivatives of the objective functions and usually provide good-quality solutions [24]. In [34], a GA (Genetic Algorithm [25]) based method is proposed to pack the irregular objects in a stock sheet with fixed width without overlap in which the irregular objects are represented by identical squares based on grid approximation. In [32], a GA-based method developed based on hyper-heuristics methodology is proposed to solve two-dimensional regular and irregular packing problems. Several heuristics methods and their placement strategies are compared in [32]. In [15], a biased random-key genetic algorithm is proposed to solve bin packing problems. Two placement strategies are proposed and the impacts of the parameters used in the placement procedure are investigated. The obtained results show that placement strategies play an important role in the optimization process. In [13], a new algorithm, called guided cuckoo search, is proposed to solve the irregular cutting problem. The proposed approach consisted of two phases. The first phase is responsible for minimizing the sheet length while the second phase is to handle the overlap minimization problem. In [21], an extended local search algorithm is proposed to solve the irregular strip packing problem. The algorithm uses two neighborhoods, one is for swapping two given polygons in a placement and the other one is for placing one polygon into a new position. The tabu search algorithm is used so as to avoid local minima. In [12], a set of robust Island Parallel Genetic Algorithms are proposed to solve one-dimensional bin packing problem. These works show that heuristic methods have become popular for irregular packing problems.

The existing results on three-dimensional packing are mainly focused on regular packing. A container-loading problem, which involves selecting how to put three-dimensional regular boxes into a given container, is studied in [27], where the container is divided into different layers. Then, boxes are put into the container through layer by layer, and a greedy randomized adaptive search procedure combined with the wall-building constructive heuristic method is proposed to guide the packing sequence. In [16], a hybrid genetic algorithm is proposed to solve this container loading problem. In [3], an approach based on beam search strategy is proposed to solve the box packing problem based on the branch-and-bound method, where only the most promising nodes at each level of the search tree are expanded. In [17], a heuristic algorithm is proposed to solve the container loading problem. Caving degree is defined as a dimensionless and normalized value of distance between the packing item and a set of packed items. This metric is used to measure how close a packing box is to those boxes which have already been packed into the container. Then, the object with the largest caving degree is chosen to pack. The advantage of this approach is that it can handle the orientation constraints. In [11], a hybrid algorithm is proposed to solve container loading problems, aiming to load a set of rectangular items (boxes) into a single rectangular large object (container), where the objective is to maximize the total volume of packed boxes. The proposed algorithm is based on bee algorithm which is a new kind of swarm intelligence. In
an integrated packing and balancing problem is formulated as a mixed-integer linear programming model. A multi-level local search heuristic is developed to solve the proposed model. All of the methods mentioned above are developed based on respective heuristic approaches to guide the packing sequence.

However, for the three-dimensional irregular packing problem, the available results are very rare. In [8], a unified framework is proposed to transfer the general packing problem (two-dimensional or three-dimensional) into an integer optimization problem in which constraints are imposed to avoid overlap. However, the number of constraints grows exponentially. Thus, the transformed integer optimization problem may become intractable even for packing only a couple of items.

It is experimentally proved that these population-based algorithms are more competitive and efficient in searching local optimum in packing problems in general. Firefly Algorithm (FA) is a novel algorithm which was firstly proposed in 2008. Since then, numerous research works and literatures (detailed in [36]) have manifested its advantages over algorithms aforementioned. Being able to successfully deal with a wide range of practical issues, FA is more advantageous in terms of less computing amount, faster calculating speed, and better accuracy and convergence [28]. Recently, FA has become an important swarm intelligence tool that has been applied in almost all areas of optimization, as well as engineering practice [38]. In many applications, FA also holds better performance. Zaman and Matin [37] have also found that FA can outperform PSO and obtained global best results. A detailed analysis has demonstrated the efficiency of FA over a wide range of test problems, such as multi-objective load dispatch problems [35]. In the classifications and clustering areas, FA also holds excellent performance. For example, Senthilnath et al. [29] provided an extensive performance study by compared FA with 11 different algorithms and concluded that firefly algorithm can be efficiently used for clustering. Furthermore, FA has also been applied to RFID network optimization [39]. Packing problem is one of the difficult discrete optimization problems which are, commonly, easily solved by intelligent algorithms. Due to superior performance of firefly algorithm, in this paper, a hybrid intelligent search algorithm based on hybridizing chaos and firefly is proposed to optimize the packing sequence. In this paper, we will study a general irregular three-dimensional packing problem. The container is divided into multiple grids and we use 1 or 0 to indicate whether a grid is occupied or not. During the packing process, each item is put through possible rotations so as to minimize the current occupied space. To validate our proposed method, several randomly generated scenarios are studied. The simulation results demonstrate that our proposed algorithm is efficient and effective.

The remainder of the paper is organized as follows. In Section 2, we introduce the irregular three-dimensional packing problem. In Section 3, three-dimensional grid representations of the container and the object are proposed and a new hybrid firefly algorithm to guide the packing sequence is derived. Several examples are presented in Section 4. Section 5 concludes the paper.

2. Problem statement. Let $O = \{o_1, o_2, \ldots, o_n\}$ denote a set of $n$ three-dimensional objects (regular or irregular). All of these objects need to be packed into a container for shipping. During the packing process, these objects can be rotated. We want to pack these objects into a container so as to minimize the occupied space. If $o_1, o_2, \ldots, o_n$ are boxes, the problem is the well-known container loading problem [3]. Here, we will consider a more general case, in which, $o_1, o_2, \ldots, o_n$, can be any geometric shape.
In applications, some objects can be loaded with arbitrary rotations, some can only be rotated with given orientations, and some are not allowed to rotate. For example, in the garment industry, only a few degrees of rotation are permitted. It is reported in [32] that rotation in multiples of 5 degrees for 2D objects shows only very marginal improvement compared to that in 90 degrees, while the multiples of 5 degrees result in extra computational load. Thus, in this paper, we only consider the limited rotation with 90 degrees. The parameters $R = (r_1, r_2, \cdots, r_n)$ will be introduced to control the number of rotations. For example, if the object $o_i$ is not allowed to be rotated, $r_i = 0$. Otherwise, $r_i$ is the number that represents the order for the object $o_i$ to rotate before packing into the container.

2.1. Geometry representation. To proceed further, we need to define a data structure for the container and the objects. This will allow us to estimate the occupied space and avoid overlap among objects. To achieve this aim, we introduce the raster method [7]. The raster method is an approach that divides the continuous space into discrete scopes. Thus, the geometric information of the object can be represented by a three-dimensional matrix.

Through introducing the raster technique, the container and the objects can be divided into a large set of boxes. The smaller the size is, the better the results are obtained. However, small size of box will increase the dimension of the search space, and thus, increases the complexity of the problem. In practice, it is usually selected based on the specific problems. Figure 1 shows how to use the raster technique to pack a cylinder into a box. In this figure, each grid is marked as either 0 or 1, where 1 means that the corresponding grid is occupied while 0 indicates that it is free. From this figure, we can also that the smaller the grid, the less the waste is.

Based on the raster model, an object can be represented through three-dimensional array. The box envelop of an object can be represented as a three-dimensional
matrix $O_i$ with the element $o_{xyz}^i$, where $x \in \{1, \cdots, O_a^i\}$, $y \in \{1, \cdots, O_b^i\}$, $z \in \{1, \cdots, O_c^i\}$, $O_a^i$, $O_b^i$ and $O_c^i$ are the numbers of grids of the enclosing box in length, width and height. In order to denote the object, as demonstrated in Figure 1, if the part of the object occupies the grid numbered $(x, y, z)$, the corresponding matrix element is set as $o_{xyz}^i = 1$, otherwise $o_{xyz}^i = 0$.

The smaller the box unit, the more accurate the approximation of the object. However, the smaller unit box needs a larger amount of units which leads to heavy computational burden. In practice, we can use a two-stage method, which will be further discussed in Section 3 to get best compromise. Box envelop, the minimal box covering the irregular object, gives us insight on how to fast load object to rough localization, and the particular elements of matrix contribute to move the box covering the irregular object, gives us insight on how to fast load object to accurate position. Note that all objects includes rotated objects should be represented three-dimensional raster matrixes before being packed into the container.

### 2.2. Mathematical formulation.

Similar to the object geometry representation, the container $C$ can be represented as the matrix with elements $C_{xyz}$ either 0 or 1, where $x \in X = \{1, \cdots, \bar{C}_a\}$, $y \in Y = \{1, \cdots, \bar{C}_b\}$, $z \in Z = \{1, \cdots, \bar{C}_c\}$, $\bar{C}_a$, $\bar{C}_b$ and $\bar{C}_c$ are the number of grids of the container in length, width and height, respectively. Now our problem can be mathematically stated as follows:

\[
\min_{i, x, y, z} \max_n d_{xyz}^i(x'_i, y'_i, z'_i, r_i) \tag{1}
\]

subject to

\[
\sum_{i=1}^n d_{xyz}^i(x'_i, y'_i, z'_i, r_i) \leq 1, \quad \forall x'_i \in X, y'_i \in Y, z'_i \in Z \quad \tag{2}
\]

\[
d_{xyz}^i(x'_i, y'_i, z'_i, r_i) \in \{0, 1\}, \tag{3}
\]

\[
r_i \in \{0, 1, \cdots, r_{max}\}, \tag{4}
\]

where

\[
d_{xyz}^i(x'_i, y'_i, z'_i, r_i) = \begin{cases} 
1, & \text{if the object } i \text{ is placed with its front-left-bottom corner} \\
0 & \text{otherwise}
\end{cases} \quad \text{at position } (x'_i, y'_i, z'_i) \text{ with rotation } r_i \text{ occupied the grid } (x, y, z);
\]

Here the variables are $r_i$ and $(x'_i, y'_i, z'_i)$. The constraint (2) is introduced to prevent the overlap between two objects in the container. (4) means that only $r_{max}$ rotations should be considered. The objective function (1) is to minimize the height of the container to be occupied. Let this problem be referred to as Problem (P).

### 3. Solution approach.

In this section, we will propose a solution strategy to solve Problem (P). Solving Problem (P) directly is very difficult because constraints are strict and number of variables is huge. For example, for a container, if we split the container as $100 \times 100 \times 100$ grids and consider 6 possible rotations for each object, the dimension of the corresponding problem will be $6 \times 10^6$. Thus, solving this small scale problem will cost unacceptable CPU time. After careful examination, we can find that once we give the rule to load the objects into the container, then the problem will be transferred to optimize the sequence of the objects loading into the container and the corresponding rotations. This is a typical scheduling problem, some heuristic methods are proposed to solve scheduling problem in [19],[9],[22].
In view of this, we will develop a heuristic approach to optimize this loading sequence and rotations. At first, we develop a hybrid firefly algorithm to fulfill this task. Based on the sequence and rotation vector, a two-stage placement method is designed to pack the objects into the container. As mentioned above, the shape matrix of an object is determined by the corresponding rotation index.

4. **Firefly algorithm.** FA is a population-based algorithm developed by simulating the social behavior of fireflies. It uses three idealized rules to search optimal solution: (i) the fireflies are attracted to other fireflies regardless of their sex; (ii) the attractiveness between two fireflies is proportional to the higher brightness and decreases with Cartesian or Euclidean distance between them; (iii) the brightness of a firefly is determined by the fitness of objective. Specially, the brightness of a firefly can be considered to proportional to fitness for maximization, on the contrary, the firefly with lowest fitness has the brightest of all fireflies for minimization.

As a population-based algorithm, FA is made up of a number of fireflies $X_i, i = 1, \ldots, \text{pop}$, with brightness $B(X_i)$ which is related to the fitness of $X_i$. In the standard firefly algorithm, the light intensity $I()$ (apparent brightness) of a firefly is proportional to the value of fitness function $I(X) \propto \text{fitness}(X)$, while the light intensity $I(d)$ varies according to the distance as followed $I(d) = I_0 e^{-\gamma d^2}$, where $I_0$ represents the light intensity of the source, and $\gamma$ is the light absorption coefficient. The attractiveness of fireflies is proportional to their light intensities $I(d)$, thus the attractiveness $\beta$ can be defined as:

$$\beta(d) = \beta_0 e^{-\gamma d^2}$$  \hspace{1cm} (5)

where $\beta_0$ is attractiveness of firefly at $r = 0$. The distance between any two fireflies $i$ and $j$ is the Cartesian distance defined as follows:

$$d_{ij} = ||X_i - X_j|| = \sqrt{\sum_{k=1}^{d} (X_{ik} - X_{jk})^2}$$  \hspace{1cm} (6)

The movement of a firefly $i$ attracted to another firefly $j$ with higher brightness is determined by

$$X_i = X_i + \beta_0 e^{-\gamma d_{ij}} (X_j - X_i) + \alpha \epsilon$$  \hspace{1cm} (7)

where $\alpha$ is randomization parameter $\alpha$, $\epsilon$ is a random number drawn from Gaussian distribution. Note that for any two fireflies, the less bright one will be attracted by (and thus move towards) the brighter one, that is to say, a firefly will be attracted by all other brighter fireflies. The new position of a firefly is determined by three terms: the current position of the firefly, attraction to another more attractive firefly, and a random walk that consists of a randomization parameter $\alpha$ which is generated from interval $[0,1]$. If there are no fireflies brighter than a given firefly, it will be only updated using random walk. The position of fireflies can be updated iteratively until convergence to an optimal solution. Firefly algorithm has been successfully applied to solve function optimization, however, when the algorithm is used to cope with problem (P), it has to be extended and enhanced due to the complexity of the proposed problem.

5. **Hybrid firefly algorithm for three-dimensional packing problem.** In this subsection, we will integrate chaos search with firefly algorithm, developing a hybrid firefly algorithm. Then, it is extended to solve the packing problem (P). To implement chaos and firefly algorithm, we need to encode the solution as individuals
to update the them and evaluate their fitness. The procedure involves encoding, decoding, evaluating, and updating. At each iteration, the following steps will be executed:

1: Encoding of the solution: This first phase of the approach is to encode the solution of the problem, which includes the sequence and the ways for placement. An individual consists of two parts, vector of sequence and vector of rotations.

2: Decoding of the solution: The second phase decodes the solution to be used by the placement procedure, where the decoding list represents the index of the rotated object for packing.

3: Updating operation of the individual: There is a significant difference between the packing problem and continuous optimization problems. In this paper, we propose a new updating method to produce individual.

4: Fitness evaluation: The fitness of the solution represents the quality of the packing. For packing problem, maximal height of container occupied by packing objects is adopted to stand for fitness. In order to achieve the occupied height, we designed a new placement strategy on the basis of decoded solution. At first, the location of enclosure of the object is located at the initial position of the container. Then, the object will be compacted as tight as possible if only overlaps do not occur. The advantage of the two-step place is time saving on search placing position.

5.1. Encoding of the firefly individual. The original firefly algorithm is designed for solving continuous optimization problems. However, packing is a discrete optimization problem. During the search process, each solution is composed of two vectors as depicted in Figure 2. The first vector \( x_i \) of individual \( i \) represents the object’s packing sequence and the second vector \( r_i \) indicates the object’s orientation ways.

In order to use firefly algorithm to solve packing problem, smallest position value (SPV) rule [30] is employed to map continuous variables to discrete sequence. The SPV rule has already been applied to solve the scheduling problems [31]. An example is presented in Figure 2. The element of vector \( x_i \) is sorted with an ascending order, then, the sequence index of each element can be calculated according to the sorted list and the obtained sequence index is used to denote the scheduling order. In Figure 2, the smallest position value is \( x_{i3} = 0.09 \). Thus, the dimension \( j = 3 \) is assigned to be the first job in the permutation according to the SPV rule.

The placement procedure exploits these two vectors to construct a final solution. In our hybrid algorithm, each individual represents a feasible packing sequence, which consists of a permutation that represents the packing of object’s index \( (o_{i1}, o_{i2}, \ldots, o_{ij}, \ldots, o_{in}) \) obtained from \( x_i \) using SPV rule, and the rotation way vector \( (r_{i1}, r_{i2}, \ldots, r_{ij}, \ldots, r_{in}) \), where \( r_{ij} \) is the rotation index of object \( j \), \( n \) is the number of packed objects. For example, as can be seen from Figure 2, the value of \( x_{ij} \) of the third object is 0.09, which means the third object will be packed at first, in addition, the index of rotation of the third object is 1 which means that the second rotated of the third object is selected, that is \( o_{i3} = 1 \). The index of the rotation represents that the object has been rotated 90 degree along \( x \)-axis, \( y \)-axis or \( z \)-axis. Since considering all rotation cases very consume time on computation, we only select limited rotation rotations of the object. Notice that all raster matrix
of the rotated object are created in advance, we can pack the rotated object directly to the container.

5.2. Chaos search optimization. Chaos is a characteristics of some types of nonlinear systems [14]. Due to non-repetitive nature of chaos, it can carry out overall searches at higher speeds than stochastic traverse searches that are probabilistic in nature. The combination of optimization methods and chaotic systems is important to avoid the disadvantages of stochastic algorithm, which has attracted increasing interest from various fields in recent years.

In most of the existing literature, the chaos queues are generated through logistic equation, which is given as follows:

\[ h_{i,j+1} = \lambda h_{i,j}(1 - h_{i,j}), \quad i = 1, 2, \ldots, n \]  

(8)

where \( h_{i,j} \) denotes the \( i \)th chaotic variable and \( j \) represents the \( j \)th iteration. \( h_{i,j} \) is distributed in the range \((0, 1)\), the sequence of \( h_{i,j} \) owns the chaotic character when \( \lambda = 4, h_{i,j} \not\in (0.25, 0.5, 0.75) \). It is reported that chaotic maps can be used to improve the diversity of the population in firefly algorithm [10], we will combine chaos maps with firefly algorithm in this paper to enhance search performance.

5.3. Movement of firefly algorithm for packing problem. The basic firefly algorithm has been illustrated in section 4. The encoding of individuals \( i \) consists of continuous variables \((x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots, x_{in})\) and integer variables \((r_{i1}, r_{i2}, \ldots, r_{ij}, \ldots, r_{in})\), the continuous variables can be updated according to formula (7). If none individual exceeds to the current individual, then, a small random disturbance is executed for the firefly as random walk operation. For the integer variables \((r_{i1}, r_{i2}, \ldots, r_{ij}, \ldots, r_{in})\), they will become floating values after updated by (7). Since the rotation index should be an integer number between 0 and \( r_{\text{max}} \), a round-off function is introduced to transform them into integer variables.

5.4. Placement strategy. The element of decode list is represented as \( l_i = o_i + r_i \ast n \). The list includes the information of index and orientation of packing object. Such as, the index of object can be obtained as \( o_i = l_i \mod n \) and the orientation \( r_i = (l_i - o_i) / n \). Based on the packing list, the object can be loaded through special placement strategy. Bottom-Left placement is one of the best known heuristic placements in two-dimensional packing problems. The basic idea is to move the shape from the top right corner of the sheet, making successive moves of sliding as far as possible to down and as far as possible to the left until the object is placed in a stable position. Bottom-Left placement is a simple and effective approach. We
extend the approach to the three-dimensional packing problem, where the placement moves the object from front-top-right to back-bottom-left of the container.

A two-stage packing approach for two-dimensional irregular packing is proposed in [34]. This method is computationally less expensive. Packing the object directly to the final position will lead to a large computational requirement on overlap test so as to avoid a new object to occupy the space which has already existed. An effective method is to replace the object by using the enclosure box. However, the enclosure box is not accurate while the object has irregular shape. Consequently, the placement first moves the object through the enclosure box without overlap test, then moves the irregular object as tight as possible under the condition that the object does not overlap with others in the container. A simple example for two objects placement is illustrated in Figure 3.

As shown in Figure 3, the container is empty at the initial stage, the first object is moved from front-top-right to back-bottom-left in Step 2. The second object is located rough coordination similar to the first object in Step 4. The object is repeated to move to left under condition that grid unit of objects being given the value 1 should not occupy the unit of container which has been occupied. Then the algorithm tries to move the object to back, and then down until the object cannot be moved any more in Step 5.

The procedure is detailed in Algorithm 2, which is similar to a wall-building method. For instance in Figure 3, the objects are moved and packed to the bottom of container. In particular, the objects are packed from left to right to structure a row at first. Then, the later objects are moved and packed to establish a new row. The procedure is repeated until the new row exceeds the the height of container. As a result, these rows consist of a wall is named a layer in Algorithm 1. The
variables $C_x, C_{y\text{bottom}}$, and $C_z$ represent the coordinations in the container for new object to place, $C_{y\text{bottom}}$ and $C_{y\text{top}}$ are bottom and top of the current layer. $o_i^x, o_i^y, o_i^z$ represent the width, height and length of the packing object $i$, and $w$ is the width of the current wall. The occupied volume $V$ for parking is is the volume that is calculated by multiplying the length occupied, width and height of container.

**Algorithm 1** Placement Procedure

**Input:**
- The decoded packing list $L_n$ and list length $n$;

**Output:**
- The occupied height $h$, efficiency $e$;

1. Initialize container, $\{C_x, C_{y\text{top}}, C_{y\text{bottom}}, C_z\} = \{0, 0, 0, 0\}$;
2. for $i=1$ to $n$ do
3. Pick $o_i$ from the packing list;
4. Move the selected object $o_i$ from front-right-top corner to back-bottom-left corner;
5. if the object $o_i$ can be placed at the current row then
6. Compact the object $o_i$ to back-bottom-left as tight as possible;
7. Update parameters $C_x = C_x + o_i^x$, $C_{y\text{top}} = \max\{C_{y\text{top}}, C_{y\text{bottom}} + o_i^y\}$, $w = \max\{w, o_i^z\}$;
8. else
9. if the object $o_i$ can be placed at the above row then
10. Add a new row, update parameters $C_x = o_i^x$, $C_{y\text{bottom}} = C_{y\text{top}}$, $C_{y\text{top}} = C_{y\text{bottom}} + o_i^y$;
11. Place the object at the above new layer, update width of current layer;
12. Update parameters $w = \max\{w, o_i^z\}$;
13. else
14. Add a new layer;
15. Update parameters $C_x = o_i^x$, $C_{y\text{bottom}} = 0$, $C_{y\text{top}} = o_i^y$, $C_z = C_z + w$, $w = o_i^z$;
16. Move the object to back-bottom-left as tight as possible;
17. Update parameters $C_x, C_{y\text{top}}, C_z$;
18. end if
19. end if
20. end for
21. Calculate the occupied capacity $V = \bar{C}_a \times \bar{C}_b \times C_z$ and efficiency $E$
22. return Occupied height $h$, efficiency $e$

5.5. Hybrid strategy and algorithm architecture. The effectiveness of the meta-heuristic algorithms is influenced by two major factors: exploration and exploitation. Exploitation involves a local search in the vicinity of the current solution, while exploration involves exploring the global search space. The proper balance between two factors affects the performance of the algorithm.

The basic firefly has a few disadvantages in the global searching, such as slow convergence speed, premature convergence. The hybrid strategy is capable of combining the advantages of different algorithms to enhance optimization performance. The new hybrid chaos firefly algorithm makes use of chaos search mechanism to increase diversity. At first, chaos is adopted at initialization to increase the diversity
Initialize Parameters:
- \( f(x) \): the efficiency of packing problem
- \( O \): the index vector of object (Decision variable)
- \( R \): the orientation vector of object (Decision variable)
- \( Y \): Number of solution

Initialize fireflies:
For \( i = 1 \) to \( Y \) do
- Randomly generate pack sequence and orientation vector
- Evaluate the solution by placement procedure algorithm

Improve new solution:
For \( i = 1 \) to \( Y \) do
- If exists individual \( j \) superior to individual \( i \) Then
  - perform movement for individual \( i \)
- else
  - perform random walk for individual \( i \)
- Evaluate the solution by placement procedure algorithm

If the new solution better than the best individual
- Update best individual
- Execute simulated annealing for best individual
- If stopping criteria satisfy
  - Terminate computation
  - Output solution

Y
N

If the best solution do not update after some step
- Use chaos process to update parameters

N

To enhance the exploitation ability of firefly algorithm, simulated annealing (SA) [20] is also integrated to form a hybrid algorithms. At each iteration, the best individual is reserved. It is note that can ensure the algorithm convergence through reserved best individual. When the best individual isn’t updated after a certain number of iterations, simulated annealing process will be performed on the all individuals to search better solution. Simulated annealing is selected because it is intensified based on neighborhood search. In order to implement the neighborhood search, the algorithm random selects two object \( o_i, o_j, i, j \in \{1, \cdots, n\} \) and sweep their packing index and rotation index. Then, one object is random selected to initialize its rotation index between 0 and \( r_{max} \). These approaches are integrated to develop a new hybrid firefly algorithm, the detail of the proposed algorithm is described in Figure 4.

5.6. Complexity and convergence analysis. Space Complexity: In our algorithm, two main memorizers are needed. The first one is matrix memorizer, which has \( O(n \times r_{max} \times O^a \times O^b \times O^c + \bar{C}_a \times \bar{C}_b \times \bar{C}_c) \), where \( O^a, O^b, O^c \) are enclosing box sizes of the maximal object, \( \bar{C}_a, \bar{C}_b, \bar{C}_c \) are size of the container, and \( n \) and \( r_{max} \) are the numbers of objects and orientations. The second one is fireflies memorizer, which has \( O(m \times n) \), where \( m \) is number of fireflies.
Computational Complexity: The main time complexity lies in the placement algorithm which needs \( O(N \times O^* \times O^* \times O^2) \) time to test overlap. The main algorithm needs \( O(M \times \text{maxgen} \times N \times O^* \times O^* \times O^2) \), where \( M \) is the number of population, \( N \) is the number of the objects, and \( \text{maxgen} \) is the number of iterations.

Next, we proof the convergence of the proposed approach. The conclusions of random search algorithm introduced by Back Thomas in [4] are used in the section to simplify the process of proof.

**Definition 1.** For any individuals \( F_1 \) and \( F_2 \), if \( F_1 \) can be translated to \( F_2 \) through the operation of FA or SA with probability that larger than 0, that is \( \text{Prob}\{\text{MTL}(F_1) = F_2\} > 0 \), then, \( F_1 \) can arrive \( F_2 \) by operators of FA or SA. \( \text{MTL}(F_1) \) is used to denote the new individual generated from \( F_1 \), and \( \text{Prob}\{\bullet\} \) indicates the probability of event \( \bullet \).

**Definition 2.** Let \( F^* \in \Omega \) denote the optimal individual of problem \( \min_{F \in \Omega} f(T) \), if \( \text{Prob}\{\lim_{t \to \infty} F^* \in S(t)\} = 1 \), the algorithm converge to global optimization with probability 1.

Particular, \( \Omega \) is the set of solutions which denote all packing sequence and rotation combination, and \( S(t) \) denotes the all individuals at step \( t \).

The conclusions of evolutionary algorithm introduced by Back Thomas in [4] indicate that the convergence will be achieved under the conditions that:

1. For each individual \( F_1, F_2 \in \Omega \), the individual \( F_2 \) can be translated from \( F_1 \) through evolution operators;
2. The population sequence \( S(0), S(1), \cdots, S(t) \) is monotonic, that is, for any \( \forall t, \min\{f(F(t)) : F(t) \in S(t)\} \geq \min\{f(F(t+1)) : F(t+1) \in S(t+1)\} \).

**Theorem 1.** The proposed HFA converges to a global optimal solution with probability 1.

**Proof.** Without loss of generality, we assume \( F_3 \) is a new individual generated from \( F_1 \) through movement of firefly algorithm. If \( \text{prob}\{T(F_3) = F_2\} > 0 \), then the individual \( F_2 \) can be translated from the individual \( F_1 \). SA random selects two object and sweep their packing index and rotation index. Further, one object is random selected to initialize its rotation index between 0 and \( r_{\text{max}} \). Thus, the probability of that individual \( F_3 \) become any individual \( F_2 \) is \( \frac{1}{n!r_{\text{max}}^n} \) through one SA operation. \( r_{\text{max}} \) denotes the rotation number of objects and \( n \) is the number of objects. Obviously, \( \text{prob}\{T(F_3) = F_2\} = \frac{1}{n!r_{\text{max}}^n} > 0 \) is satisfied. Further, the individual \( F_2 \) can be translated from the individual \( F_1 \).

Secondly, elitism preserving strategy is integrated in the proposed algorithm, thus, the best individual in population at step \( t + 1 \), \( \{p(t+1, i) | i = 1 \sim N\} \), is not inferior to the best individual in \( \{p(t, i) | i = 1 \sim N\} \), where \( t = 1, 2, \cdots \). Therefore, \( \min\{f(T(t)) : T(t) \in S(t)\} \geq \min\{f(T(t+1)) : T(t+1) \in S(t+1)\} \), that is, population sequence \( S(0), S(1), \cdots, S(t) \) is monotonic. As mentioned above, the proposed HFA is proofed to converge to global optimal solution with probability 1. This completes the proof.

Note that proposed HFA can converge to a global optimal with determined evaluation function. However, in this paper we only consider several orientation of the object. The proposed HFA only searches optimal packing and orientation sequence for placement procedure to pack objects.

5.7. **Extension to parallel implementation.** Packing a large number of objects costs significant computational time. To solve the problem, parallel implementation...
6. Experiments and discussion. In this section, we will conduct extensive experiments to evaluate the performance of the proposed hybrid algorithm. The presented algorithms are implemented in the C++ language and compiled using VC++ 6.0 compiler. To our best knowledge, there are no benchmarks on three-dimensional irregular packing problems available in literatures. Hence, we construct three groups irregular objects, including 9 instances of irregular objects, to evaluate the proposed packing algorithm. The characteristics of the instances are shown in Table 1. Specially, the first group (instance 1- instance 3) only contains cylinders. The second group (instance 4- instance 6) consists of cylinders and multiple types irregular complex structures. The third one (instance 7- instance 9) includes irregular complex structures composed by multi-boxes and cylinders. In addition, the number column represents the amount of objects in the corresponding instance. Note that the value of rotation indicates the number of objects rotated from the original object, such as, if the value is 0, it means the object can not be rotated. The scale of objects states the objects are not larger than the corresponding scope.

6.1. Configuration of hybrid algorithms. In order to evaluate and compare the performance of proposed algorithms, a novel packing method, named particle swarm optimization [18] guided 3-dimensional bin packing approach is also modified to solve the instances in Table 1. We named the proposed approach as hybrid firefly algorithm (HFA). A classic genetic algorithm [5] is also extended to optimize the problem as compared algorithm. Furthermore, we named the approach without simulated annealing operator as firefly algorithm (FA). All these population based algorithm use the same number of individuals to search solution for same case. The possibility of acceptance of worse solution is \( p = e^{\Delta f / T} \), where \( \Delta f \) is the reduction
Table 2. Parameters of the hybrid firefly algorithm

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>population size</td>
<td>number \times (1+r_{max})</td>
</tr>
<tr>
<td>$T_0$</td>
<td>0.064</td>
</tr>
<tr>
<td>Temperature update ratio</td>
<td>1.6</td>
</tr>
<tr>
<td>Iteration</td>
<td>300</td>
</tr>
</tbody>
</table>

of fitness value, $T_k = T_{k-1}/1.6$ is the temperature of $k^{th}$ iteration. The parameters of HFA and CFA are shown in Table 2.

It is well-known that population size plays a key factor on the performance of population-based heuristic optimization. We use more populations for more complex problems, in table 2, $n$ represents the number of the objects which will be packed into the container. $r$ denotes the rotation number of object. For each instance, all algorithms are set identical population size and iteration.

6.2. Visual packing example. We first evaluate the problem of packing various size cylinders into a container. The example is motivated by industrial applications. Packing cylinders arises in nuclear physics, distillation and gas absorption, casting techniques, and granular materials [8]. The number of cylinders and their characteristics are defined as instance 1 in Table 2. The instance contains 50 vertically oriented cylinders, while rotation is not considered because it is not permitted to rotate the cylinder in most cases. In order to compare the packing results obtained by HFA with others, we select the best one from the first generation population of HFA. This is because the worst one is too poor to be used for comparison. The visual results of a random sequence packing and an optimized sequence packing are presented in Figure 5. Due to the reason that the performance of random sequence is very low. The best one of initial populations of HFA is selected as compared object. Specially, the height obtained by random sequence is 136, while the height obtained by the optimized sequence is 116. It nearly decrease 15% height using HFA to search pack sequence. Obviously, the optimized approach can reduce the occupied capacity significantly.

Secondly, we evaluate the packing result for instance 6 which contains 100 irregular objects. In this case, we do not consider rotation. A random packing list and optimized list obtained by HFA are use to pack the objects. The visual result are shown in Figure 6. To elaborate, the height obtained by random sequence is 521 and the height obtained by the optimized sequence is 449. It almost cuts down 14% height by optimized packing sequence.

Lastly, we compare the packing result for instance 9 with considering rotation. In this case, 60 irregular object are packing into a container. We also use random packing list and rotation vector to place the objects as well as optimized packing list and rotation vector. Figure 7 demonstrates the packing result. Note that the height obtained by random sequence is 276 and the height obtained by the optimized sequence is 227. The result shows consistent performance with the above two examples. Next, we will compare the performance of proposed algorithm with other algorithms in the following section.

6.3. Convergence of the algorithms. In this section, we will evaluate the convergence of the proposed algorithm as well as compared algorithms. To evaluate the performance of the proposed algorithm, three algorithms including GA, PSO, and
FA are selected for comparison. As mentioned above, GA [16] is a classic optimization method to pack cargos. PSO is a popular heuristic algorithm that is widely used in many domains including packing problems [18]. Since FA is extended from PSO, the encoding of individual also can be used by PSO to solve the packing problem in this paper. Furthermore, SPV and round-off are also adopted for PSO to convert continuous variables to integer variables. The population of four algorithms are set as the same number. The crossover and mutation rate of GA are set as 0.8 and 0.06 respectively. The parameters of PSO are set as \( w = 0.7, r_1 = r_2 = 1.49 \). The parameters for our hybrid firefly algorithm are given in Table 2. We run GA, PSO, FA and HFA 10 times for 1,4,6,9 case in Table 1, respectively. The selected four cases denote different types and scales. The average fitness of every iteration for the instances are calculated. Then, the convergence of four approaches for four cases are presented in Figure 8.

It is obviously inferred from Figure 8 that the fitness of four algorithms decrease faster in earlier iterations than latter ones. This indicates that the population-based algorithm can quickly obtain reasonable solutions. Furthermore, in the most cases, the solutions obtained by HFA is better than the solutions obtained by compared algorithms after 150 iterations. This is mainly due to the reason that HFA integrated local search operator, which enhances the exploitation ability of hybrid algorithm.

6.4. **Evaluating results of irregular packing problem.** In order to evaluate the results of algorithms for the instances in detail, two metrics contains height and efficiency are used to measure results. The metric height denotes the maximal occupied height after packing objects while efficiency represents the ratio of
Figure 6. The comparison packing result of instance 6 without rotation

Table 3. The maximal height and the efficiency achieved by three algorithms in 10 runs

<table>
<thead>
<tr>
<th>Instance</th>
<th>GA Height</th>
<th>GA efficiency</th>
<th>PSO Height</th>
<th>PSO efficiency</th>
<th>FA Height</th>
<th>FA efficiency</th>
<th>HFA Height</th>
<th>HFA efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>122</td>
<td>52.5%</td>
<td>120</td>
<td>55.4%</td>
<td>117</td>
<td>56.8%</td>
<td>120</td>
<td>55.4%</td>
</tr>
<tr>
<td>2</td>
<td>169</td>
<td>68.0%</td>
<td>171</td>
<td>67.2%</td>
<td>170</td>
<td>67.6%</td>
<td>167</td>
<td>68.8%</td>
</tr>
<tr>
<td>3</td>
<td>222</td>
<td>64.0%</td>
<td>219</td>
<td>64.9%</td>
<td>221</td>
<td>64.4%</td>
<td>219</td>
<td>64.9%</td>
</tr>
<tr>
<td>4</td>
<td>210</td>
<td>50.1%</td>
<td>207</td>
<td>50.8%</td>
<td>215</td>
<td>48.9%</td>
<td>198</td>
<td>53.1%</td>
</tr>
<tr>
<td>5</td>
<td>366</td>
<td>53.3%</td>
<td>363</td>
<td>54.3%</td>
<td>365</td>
<td>53.4%</td>
<td>356</td>
<td>55.4%</td>
</tr>
<tr>
<td>6</td>
<td>446</td>
<td>51.8%</td>
<td>439</td>
<td>52.0%</td>
<td>448</td>
<td>51.5%</td>
<td>442</td>
<td>52.2%</td>
</tr>
<tr>
<td>7</td>
<td>160</td>
<td>59.9%</td>
<td>160</td>
<td>59.9%</td>
<td>161</td>
<td>59.6%</td>
<td>157</td>
<td>61.1%</td>
</tr>
<tr>
<td>8</td>
<td>230</td>
<td>61.0%</td>
<td>231</td>
<td>60.7%</td>
<td>234</td>
<td>59.9%</td>
<td>225</td>
<td>62.3%</td>
</tr>
<tr>
<td>9</td>
<td>310</td>
<td>58.9%</td>
<td>308</td>
<td>59.3%</td>
<td>304</td>
<td>60.1%</td>
<td>304</td>
<td>60.1%</td>
</tr>
</tbody>
</table>

total volume of the objects /occupied volume of container. We compare the results obtained by the four algorithms 10 times for 9 instances without considering rotation. The configuration of the algorithms are set on the basis of Table 2. Then, the best and average results obtained by each algorithm are shown in Table 3. As can be observed from Table 3, HFA obtains better values for 7 cases, and obtains worse values for only 2 cases than compared algorithms. Note that the results only select best one from the solutions obtained by 10 run of each algorithm. Moreover, the statistical results are detailed in Table 4. In Table 4, each row shows the best and average values obtained by three algorithms for the instances. Column 4
A HYBRID CHAOS FIREFLY ALGORITHM

Figure 7. The comparison packing result of instance 8 with rotation

Table 4. The statistical performance of the algorithm without rotation

<table>
<thead>
<tr>
<th>Ins</th>
<th>GA</th>
<th>Avg</th>
<th>Stdev</th>
<th>Best</th>
<th>Avg</th>
<th>Stdev</th>
<th>Best</th>
<th>Avg</th>
<th>Stdev</th>
<th>Best</th>
<th>Avg</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>120</td>
<td>122.1</td>
<td>2.172</td>
<td>120</td>
<td>121.3</td>
<td>1.341</td>
<td>117</td>
<td>120.8</td>
<td>2.049</td>
<td>120</td>
<td>120.3</td>
<td>0.547</td>
</tr>
<tr>
<td>2</td>
<td>171</td>
<td>172.5</td>
<td>2.918</td>
<td>171</td>
<td>172.1</td>
<td>2.121</td>
<td>170</td>
<td>172.5</td>
<td>3.140</td>
<td>167</td>
<td>170</td>
<td>2.387</td>
</tr>
<tr>
<td>3</td>
<td>220</td>
<td>224.6</td>
<td>2.671</td>
<td>219</td>
<td>223.1</td>
<td>1.923</td>
<td>221</td>
<td>222.3</td>
<td>2.100</td>
<td>219</td>
<td>221.2</td>
<td>1.483</td>
</tr>
<tr>
<td>4</td>
<td>210</td>
<td>219.8</td>
<td>3.019</td>
<td>207</td>
<td>218.6</td>
<td>2.074</td>
<td>215</td>
<td>220.9</td>
<td>8.648</td>
<td>198</td>
<td>215.2</td>
<td>6.638</td>
</tr>
<tr>
<td>5</td>
<td>362</td>
<td>371.1</td>
<td>4.017</td>
<td>363</td>
<td>371.4</td>
<td>6.058</td>
<td>356</td>
<td>370.6</td>
<td>10.425</td>
<td>356</td>
<td>368.8</td>
<td>6.025</td>
</tr>
<tr>
<td>6</td>
<td>445</td>
<td>465.2</td>
<td>8.423</td>
<td>439</td>
<td>461.2</td>
<td>8.820</td>
<td>448</td>
<td>459.4</td>
<td>9.597</td>
<td>442</td>
<td>452</td>
<td>4.264</td>
</tr>
<tr>
<td>7</td>
<td>162</td>
<td>168.1</td>
<td>9.150</td>
<td>160</td>
<td>168.2</td>
<td>9.517</td>
<td>161</td>
<td>167.6</td>
<td>8.961</td>
<td>157</td>
<td>162.9</td>
<td>4.868</td>
</tr>
<tr>
<td>8</td>
<td>232</td>
<td>245.1</td>
<td>6.901</td>
<td>231</td>
<td>242</td>
<td>7.615</td>
<td>254</td>
<td>244.2</td>
<td>3.421</td>
<td>225</td>
<td>234.2</td>
<td>2.863</td>
</tr>
<tr>
<td>9</td>
<td>311</td>
<td>314.6</td>
<td>6.119</td>
<td>308</td>
<td>313.8</td>
<td>5.354</td>
<td>304</td>
<td>316.2</td>
<td>7.190</td>
<td>304</td>
<td>307.6</td>
<td>3.050</td>
</tr>
</tbody>
</table>

Named stdev refers to standard deviation of height, which reflects the robustness of performance of algorithms.

It can be inferred from Table 4, HFA is significantly better than all the other algorithms on average results. The average height obtained by HFA superior to other three algorithms for 9 instances. Further, HFA has excellent stability since it achieves lower standard deviation for 7 cases over 9 instances, that is almost 78%. The experiment results demonstrate that our proposed algorithm is effective and robust. This is mainly due to the reason that hybrid operators can effectively improve global and local search ability through integrated SA.

Next, we validate the performance of algorithms with extensive experiments. In the following experiment, we consider packing with/without compacting operator, as well as packing with/without rotation. The algorithm parameters are configured...
in table 2. Note that instances 1, 2, and 3 are ignored to rotated because they are all cylinder. We run hybrid chaos firefly algorithm 10 times to obtain average height of each instances. Table 5 retains the results for enclosure (no compacting operator), no-rotation, and rotation cases. To elaborate, each row in Table 5 states the result of an instance. Column enclosure indicates that placement algorithm only place objects refer to their enclosure rather than compact them. Note that enclosure of object denotes a minimal box covered the object as defined above. The third column refers to the proposed algorithm without rotation. Obviously, with compact operator, the performance of hybrid algorithm is improved consistently. Among the 9 cases, the improvement of 9 instances higher than 3%, and the highest improvement is No.8 instance, its improvement is about 4.80%. This explicitly show that the performance of the proposed approximation exceeds enclosure significantly. The results demonstrate that the irregular approximation approach is effective in term of volume saving.

The fifth column in Table 5 refers to the results considering rotation obtained by HFA. The sixth column represents the enhancement ratio of rotation ones compared to ones without rotation for instances in Table 1. The result show that rotation can decrease height consistently. Among which, the best instance is No.7, which enhances performance about 3.99%. One explain is that rotation is contribute to compact objects more tightly, further result in reducing occupied volume. However, on the contrary, the algorithm has to consume more computational power if rotation is considered since rotation enlarges search space significantly. Hence, we need to use more individuals to solve the packing problem.
Table 5. Comparison between the proposed approach and placement strategy

<table>
<thead>
<tr>
<th>Instance</th>
<th>enclosure without rotation</th>
<th>enhance(%)</th>
<th>with rotation</th>
<th>enhance(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>124.2</td>
<td>120.3</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>2</td>
<td>173.4</td>
<td>170.0</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>225.3</td>
<td>221.2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>225.2</td>
<td>215.2</td>
<td>4.44%</td>
<td>210.2</td>
</tr>
<tr>
<td>5</td>
<td>381.6</td>
<td>365.5</td>
<td>4.40%</td>
<td>358.3</td>
</tr>
<tr>
<td>6</td>
<td>466.3</td>
<td>452.0</td>
<td>3.16%</td>
<td>445.9</td>
</tr>
<tr>
<td>7</td>
<td>170.9</td>
<td>162.9</td>
<td>4.68%</td>
<td>156.4</td>
</tr>
<tr>
<td>8</td>
<td>246.0</td>
<td>234.2</td>
<td>4.80%</td>
<td>230.6</td>
</tr>
<tr>
<td>9</td>
<td>319.6</td>
<td>307.6</td>
<td>3.90%</td>
<td>301.7</td>
</tr>
</tbody>
</table>

7. Conclusion. Three-dimensional irregular object packing has been studied in the paper. Firstly, a three-dimensional grid approximation approach is proposed to represent the shape of a three-dimensional object, which can specify any shape to three-dimensional matrix, and is easy to check overlap. Secondly, a heuristic search approach based on placement strategy is developed to optimize the permutation of the object for packing. The placement is divided into two-stage which searches the rough position in the container for the object at first, then packs the object as tight as possible under non-overlap condition. Since this is a new problem in the literature, benchmark problems are not available, and thus new benchmark data set are presented for the problem. The experiments in the paper illustrate our proposed algorithm performance superior to comparison algorithm.

There is significant scope for more research on this problem. How to improve the quality of packing under considering stability, and how to apply it to engineering projects including irregular pipe packing still call for further investigation.

REFERENCES


Received March 2017; 1st revision January 2018; 2nd revision April 2018.

E-mail address: zcxonline@126.com
E-mail address: lydijianglin@gmail.com
E-mail address: K.L.Teo@curtin.edu.au